Hindawi Publishing Corporation
The Scientific World Journal
Volume 2014, Article ID 735457, 14 pages
http://dx.doi.org/10.1155/2014/735457

Research Article

Adaptive Broadcasting Mechanism for Bandwidth Allocation in Mobile Services

Gwo-Jiun Horng,[1] Chi-Hsuan Wang,[2] and Chih-Lun Chou[3]

[1] *Department of Computer Science and Information Engineering, Southern Taiwan University of Science and Technology, Tainan 710, Taiwan*
[2] *Department of Electrical Engineering, National Cheng Kung University, Taiwan*
[3] *Department of Information and Telecommunications Engineering, Ming Chuan University, Taiwan*

Correspondence should be addressed to Gwo-Jiun Horng; grojium@gmail.com

Received 5 February 2014; Revised 15 April 2014; Accepted 15 May 2014; Published 26 June 2014

Academic Editor: Wen-Chiung Lee

This paper proposes a tree-based adaptive broadcasting (TAB) algorithm for data dissemination to improve data access efficiency. The proposed TAB algorithm first constructs a broadcast tree to determine the broadcast frequency of each data and splits the broadcast tree into some broadcast wood to generate the broadcast program. In addition, this paper develops an analytical model to derive the mean access latency of the generated broadcast program. In light of the derived results, both the index channel's bandwidth and the data channel's bandwidth can be optimally allocated to maximize bandwidth utilization. This paper presents experiments to help evaluate the effectiveness of the proposed strategy. From the experimental results, it can be seen that the proposed mechanism is feasible in practice.

1. Introduction

Mobile web services are a new generation of web services accessible to mobile clients through the air in support of anytime-and-anywhere access to services [1, 2]. Furthermore, owing to the characteristics of wireless environments including device mobility, scarce bandwidth, and limited battery power, accessing services in wireless-oriented service environments has become an emerging challenge to the data-management and telecommunication communities [3].

In essence, there are two fundamental modes for data-service dissemination in a wireless region: the broadcasting mode and the on-demand mode [4]. In a broadcasting mode, data is broadcast periodically to mobile devices according to a broadcast program in the region [5–14]. To fetch a data record, mobile clients have to wait until the target data appears on the broadcast channel. In this way, a broadcast-based system can serve thousands of mobile users simultaneously, since the broadcast cost is identical regardless of the number of users. The other data dissemination mode is the on-demand mode. This mode is similar to the traditional client-server approach. In the on-demand mode, a mobile node first sends its query on an uplink channel and the server sends the requested data to the client through the downlink channel. In this paper, we consider data disseminated in a broadcast-based wireless environment.

In the literature, access efficiency and energy consumption are two issues of concern in assessing the performance of wireless communication systems [4, 15]. Access efficiency can be evaluated by access time, which means the time that has elapsed from the moment a client requests data to the moment the client retrieves the target item. Energy consumption concerns the battery power consumed by the client to retrieve the requested data, and it can be quantified according to tune-in time [15], in other words, according to the amount of time the mobile device stays active "listening" to the broadcast channel.

The plain-broadcast scheme is the simplest approach to generating data-broadcast programs and has been adopted in earlier research [3, 16]. Using this approach, the server broadcasts all data records in a round robin manner. Therefore, this method is easily implemented. Furthermore, since the plain-

broadcast scheme treats all data items equally, the average waiting time for each packet of data equals half of the overall broadcast period. As a result, it is clear that this scheme is not feasible for cases in which data-access frequencies are not uniform.

An alternative data dissemination mechanism is the broadcast disks scheme, which permits data items to be broadcast with different frequencies [5]. This algorithm first divides data items into a few groups (i.e., disks) such that data items with similar popularity are assigned to the same disks. Afterwards, it determines the rotation speed of each disk according to the popularity of data items. In this way, one can construct a broadcast program that adjusts the trade-off between the access time of hot data and that of cold data.

In addition to access efficiency, power conservation is critical for mobile nodes owing to limited battery capacities [17–19]. To facilitate power saving, it is necessary for mobile devices to support two operation modes: the active mode and the doze mode [20]. Mobile clients normally operate in the active mode, and they can switch to the energy-saving doze mode when mobile devices become idle. Thus, keeping mobile devices in the doze mode for as long as possible could be achieved through the application of an air index technique.

By broadcasting the arrival time of data items to clients, mobile devices can stay in the doze mode until the requested data arrives. In this way, the tune-in time can be reduced to the initial index probe time plus the data-retrieval time. At present, several research efforts have addressed reducing the initial probe time [15, 21–26]. These studies complement our work in different aspects.

In this paper, we investigate the effects of data-access frequency and data size on access efficiency. And we propose the tree-based adaptive broadcasting (TAB) algorithm for skewed-data-access to generate an efficient broadcast cycle. The TAB algorithm first generates a broadcast tree to determine the broadcast frequency of each data record. After that, the broadcast tree is split into broadcast wood to balance the interbroadcast time of successive copies of data. In order to reduce the tune-in time, we further separate one individual channel from the broadcast channel to broadcast index packets.

The rest of this paper is organized as follows. Section 2 introduces related data-broadcast research. The system architecture used throughout this paper is presented as well. Section 3 discusses the proposed TAB algorithm and its role in improving data-access latency. Section 4 establishes an analytical model for optimizing index-channel and data-channel bandwidth allocation. Section 5 discusses the proposed dynamic broadcast adaptive for weight change. Section 6 discusses experiments serving to evaluate the performance of the proposed mechanism. Finally, Section 6 remarks on the conclusions drawn.

2. Related Work

This section reviews important attempts at applying data broadcasting and bandwidth allocation in mobile networks. This paper develops an analytical model to approximate the proposed TAB algorithm. This model makes it convenient to efficiently evaluate the mean access time of the generated broadcast program. Moreover, in light of the derived access time, both the index channel's optimum bandwidth allocation and the data channel's optimum bandwidth allocation are formulated.

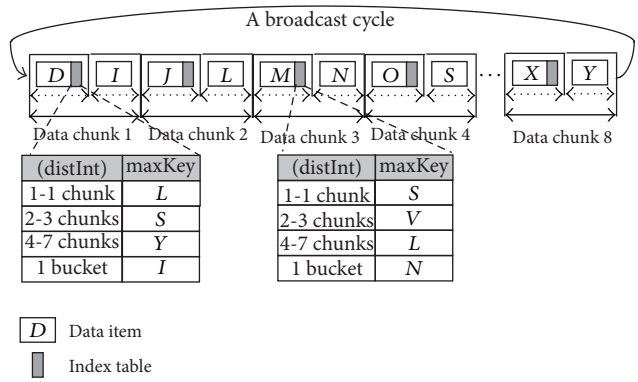

Figure 1: An example of an exponential index.

In order to assess the feasibility and efficiency of our mechanism, we conducted several experiments. Results reveal that the proposed TAB algorithm performs well in terms of data-access efficiency. Moreover, the optimum bandwidth allocation yields a significant performance improvement in tune-in time. As a consequence, it can be seen from experimental results that putting the proposed mechanism into practice is entirely feasible.

In [5], this scheme assumes that data items are of equal sizes. In terms of practicality, it is not efficient to apply the broadcast disks to the varied-size data items. Moreover, it is hard for system developers to define the similarity of data popularity so as to partition data items into disks. The determination of the relative broadcast frequency for each disk is also imprecise. In this paper, we propose a TAB algorithm for varied-size data items to tackle the above drawbacks. The TAB algorithm grows a broadcast tree to determine the broadcast frequency of each data record. After that, we split the broadcast tree into some broadcast wood with similar sizes so as to place those data items in the broadcast cycle. The details are described clearly in Section 3.

Xu et al. present exponential-index technology that enables some flexibility in trade-off between tune-in time and access latency [21]. As shown in Figure 1, the exponential index adopts a flat broadcast and disseminates data items in the ascending order of their identifiers. These researchers further group data items into a chunk and maintain one index table for each chunk. The number of entries in an index table is determined by the index base. Then the exponential-index technology can adjust the trade-off between access efficiency and energy consumption by tuning the index base and the chunk-size parameters.

Therefore, the bandwidth of an exponential index in broadcasting index information is only dominated by the index base and chunk size. The current study further examines the effects of data placement on broadcast programs and establishes an analytical model to derive the optimum bandwidth allocation for index packets and data elements.

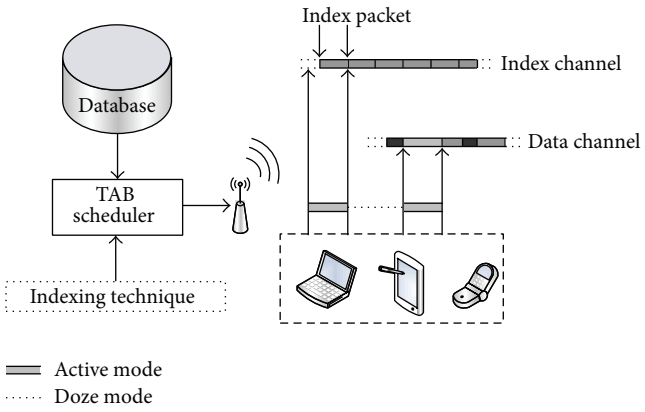

Figure 2: The sketch of our system architecture.

The performance comparison between our mechanism and the exponential index is demonstrated in Section 5.

The system architecture in this paper is depicted in Figure 2. As shown in Figure 2, the proposed TAB algorithm schedules data items in a server's database to construct a broadcast program. According to the broadcast program, the system disseminates these data records periodically through the data channel. In addition, in order to reduce power consumption, some effective indexing techniques can generate index packets. Those index packets contain information for mobile nodes, such as data identifiers and the nearest data-appearance time. Index packets are broadcast through the index channel.

On the other hand, when a user submits queries to a mobile client, the mobile equipment first fetches an index packet from the index channel to get the arrival time of the target data. Then, the mobile device switches from the active mode to the doze mode for energy savings until the target item appears on the data channel [20]. After that, the mobile client downloads the target-data transaction so as to process the user's request.

Servers' databases maintain some auxiliary information for each data item. As depicted in Table 1, each data entry consists of three attributes: the data identifier, the data-access probability, and the data size. Addressing these factors, this paper proposes an efficient TAB algorithm to generate a skewed broadcast program. Afterwards, the paper proposes an analytical model to approximate the mean access time of our mechanism. We also derive the optimum index-channel and data-channel bandwidth allocations. Details are presented in the following sections.

3. Tree-Based Adaptive Broadcasting

In this section, we propose the TAB algorithm as a way to improve the performance of existing data-broadcasting mechanisms. According to the statistical probability of data access, the TAB algorithm broadcasts a significant number of copies for popular data in a broadcast cycle to diminish the average access time. In addition, the proposed algorithm balances the interbroadcast time of successive copies of a single packet of data even though data-item sizes can vary.

Table 1: Data structure in servers' databases.

Data identifier	Access probability	Size
D_1	$\Pr(D_1)$	$S(D_1)$
D_2	$\Pr(D_2)$	$S(D_2)$
D_3	$\Pr(D_3)$	$S(D_3)$
⋮	⋮	⋮

Table 2: Notation used in the TAB mechanism.

Notation	Definition
N	The number of data items
$\Pr(D_i)$	The access probability of data item D_i
$S(D_i)$	The size of data item D_i
n_i	The broadcast frequency of data item D_i
$\mathfrak{R}$	Sapling
$\mathfrak{T}$	Broadcast tree
h	Tree height
L	The length of a broadcast cycle
B	The bandwidth of a broadcast channel
τ	Maximum tree height of a broadcast tree
w_i	The ith broadcast wood

The notation for the TAB algorithm is summarized in Table 2. Consider the case in which a server's database contains N data items for broadcasting. The data-access probability and the data size for each data item are given as well. In terms of these factors, the TAB algorithm should construct an efficient broadcast program to reduce access time. In fact, the TAB algorithm can be split into three different steps: data-item reordering, broadcast-tree construction, and wood-size equalization. The details for each step are described as follows.

3.1. Data-Item Reordering. The first step of the TAB algorithm is to sort all data records in the database by their access frequency and size. More precisely, after performing the data-item reordering, we would get a broadcast cycle $[D_1, D_2, \ldots, D_N]$ such that (1) $\Pr(D_i) \geq \Pr(D_j)$ and (2) if $\Pr(D_i) = \Pr(D_j)$, and then $S(D_i) \leq S(D_j)$ for any integers $i < j \leq N$. To reduce the average access time, it is beneficial to broadcast hotter data more frequently [27, 28]. Therefore, sorting data records from hottest to coldest can make it convenient to determine the broadcast frequency of each data item.

The second step is to determine the broadcast frequency (i.e., the number of replicates) for each data item. Note that the replication of a data record would reduce the access time of that data; however, the replication would lengthen the whole broadcast cycle and increase the access time of other records. Therefore, this paper proposes a broadcast-tree construction algorithm to balance the trade-off between these two factors.

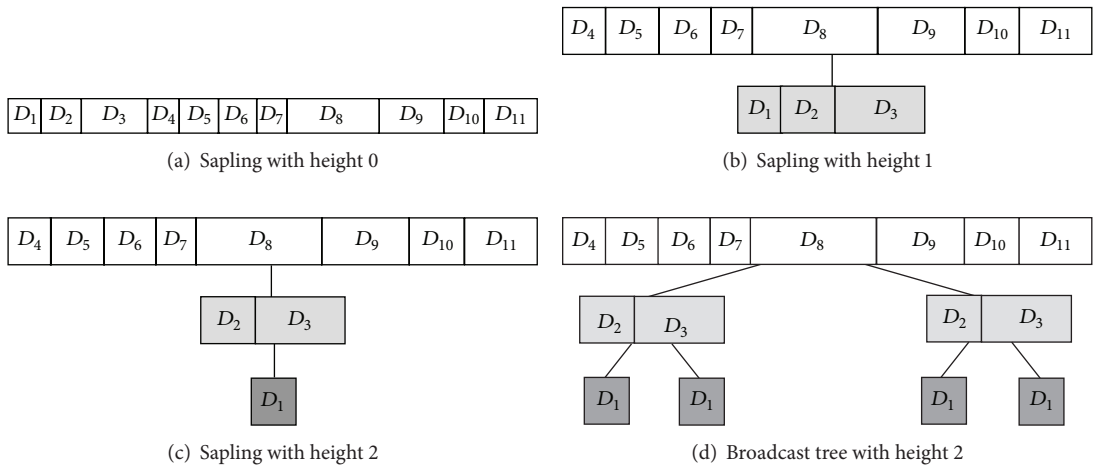

Figure 3: The scenario of the broadcast-tree construction.

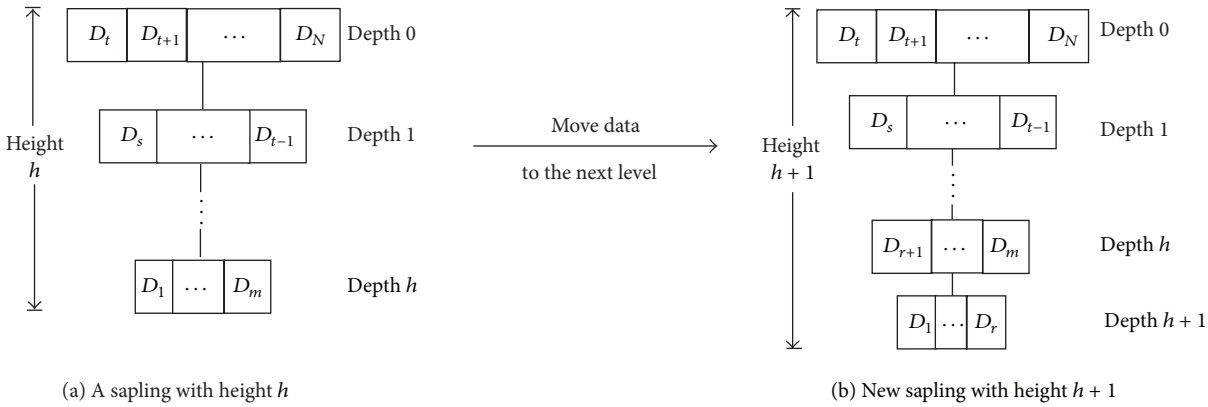

Figure 4: Move data items $\{D_1, \ldots, D_r\}$ to the next level.

As a matter of fact, the broadcast-tree construction yields broadcast trees in a top-down manner. Figure 3 presents the scenario of the broadcast-tree construction. First of all, the broadcast-tree construction starts with the sorted data from the data-item reordering (Figure 3(a)). Then, after some evaluation, the algorithm iteratively moves data items with high access probabilities to the lower level so as to double their broadcast frequencies (Figures 3(b) and 3(c)). Finally, we can get a broadcast tree by copying the nodes at each level, resulting in a full binary tree as drawn in Figure 3(d).

3.2. Broadcast-Tree Construction. Once the broadcast tree is built, the broadcast frequency n_i for each data item D_i is determined as well. The number of replicates in the broadcast tree stands for the data's broadcast frequency. Thus, take the broadcast tree in Figure 3 as an example. In this case, we have $n_1 = 4$, $n_2 = n_3 = 2$, and $n_4 = n_5 = \cdots = n_{11} = 1$. Specifically, the criterion for estimating which data items should be moved to the next level is determined by the following theorem.

Theorem 1. *Suppose that sapling $\mathfrak{R}$ of height h has m data records at depth h (as shown in Figure 4(a)). Let d_i denote the depth of data item D_i in $\mathfrak{R}$ and let $S(D_i)$ represent its data size (i.e., the broadcast frequency $n_i = 2^{d_i}$ and the broadcast length $L = \sum_{i=1}^{N} n_i S(D_i)$. Assume that the bandwidth of the data channel is B. Then the reduced average access time, achieved by moving data items $D_1, \ldots, D_r$ $(1 \le r \le m)$ to the next level $h + 1$, can be formulated by*

$$\frac{L}{2^{h+2}B}\sum_{i=1}^{r}\Pr(D_i) - \frac{1}{B}\left(\sum_{i=1}^{r}\frac{\Pr(D_i)}{4} + 2^{h-1}\sum_{i=r+1}^{N}\frac{\Pr(D_i)}{n_i}\right)\left(\sum_{i=1}^{r}S(D_i)\right). \tag{1}$$

Proof. Consider the average access time before and after moving data to the next level. Figure 4 shows that, before data are moved, one can perform an approximate calculation of the mean access time T_{BE} by using the equation

$$T_{\mathrm{BE}} = \sum_{i=1}^{N}\Pr(D_i)\left(\frac{L_{\mathrm{BE}}}{2n_iB} + \frac{S(D_i)}{B}\right), \tag{2}$$

Broadcast-Tree Construction Procedure:
(0) Initial settings: Let $h = 0$, $m = N$, $L = \sum_{i=1}^{N} S(D_i)$, and $n_i = 1$ for $i = 1, 2, \ldots, N$
(1) Let $F(r) = (L/2^{h+2})\ \sum_{i=1}^{r} \Pr(D_i) - \left(\sum_{i=1}^{r}(\Pr(D_i)/4) + 2^{h-1}\sum_{i=r+1}^{N}(\Pr(D_i)/n_i)\right)\left(\sum_{i=1}^{r} S(D_i)\right)$.
Find the cutpoint c such that $F(c) = \max_{1\le r\le m}\{F(r)\}$.
(2) **If** $F(c) \le 0$ or $h > \tau$, then **return** the expanded full binary tree $\mathfrak{T}$.
(3) **else** move the data items $\{D_1, \ldots, D_c\}$ to the next level.
(4) Set $h = h + 1$, $m = c$, $L = L + \sum_{i=1}^{c} n_i S(D_i)$, and $n_i = 2^h$ for $i = 1, 2, \ldots, c$. Then go to step (1).

Algorithm 1

where L_{BE} denotes the current broadcast length. Likewise, after data are moved, the average access time T_{AF} can be estimated by means of the equation

$$T_{\mathrm{AF}} = \sum_{i=1}^{r} \Pr(D_i)\left(\frac{L_{\mathrm{AF}}}{2\cdot(2n_i)B} + \frac{S(D_i)}{B}\right) + \sum_{i=r+1}^{N} \Pr(D_i)\left(\frac{L_{\mathrm{AF}}}{2n_iB} + \frac{S(D_i)}{B}\right), \tag{3}$$

where the latter broadcast cycle length L_{AF} is equal to $L_{\mathrm{BE}} + \sum_{i=1}^{r} n_i S(D_i)$. Therefore, we can obtain the reduced access time

$$\begin{aligned} T_{\mathrm{BE}} - T_{\mathrm{AF}} &= \sum_{i=1}^{r}\frac{\Pr(D_i)}{4n_iB}L_{\mathrm{BE}} - \left(\sum_{i=1}^{r}\frac{\Pr(D_i)}{4n_iB} + \sum_{i=r+1}^{N}\frac{\Pr(D_i)}{2n_iB}\right)\left(\sum_{j=1}^{r} n_j S(D_j)\right) \\ &= \frac{L_{\mathrm{BE}}}{2^{h+2}B}\sum_{i=1}^{r}\Pr(D_i) - \frac{1}{B}\left(\sum_{i=1}^{r}\frac{\Pr(D_i)}{4} + 2^{h-1}\sum_{i=r+1}^{N}\frac{\Pr(D_i)}{n_i}\right)\left(\sum_{j=1}^{r} S(D_j)\right). \end{aligned} \tag{4}$$

According to Theorem 1, the constancy of bandwidth B facilitates the broadcast-tree construction procedure. The broadcast-tree construction starts with the sorted data elements from the data-item reordering.

Afterwards, we use Theorem 1 to determine the optimal cutpoint c for each level and move data records $D_1, \ldots, D_c$ to the next floor so as to reduce the overall access time. In addition, note that the maximum height of the generated broadcast tree is limited by parameter τ. This factor can prevent the procedures shown in Algorithm 1 from taking too much execution time. □

3.3. Wood-Size Equalization. Once the number of duplicates for each data item is obtained, we determine the replicated data placement in the broadcast cycle. Clearly, to achieve a better performance, the interbroadcast time of successive copies of data should be the same. However, it is known that such an optimum placement problem associated with the variant data sizes is an NP-complete problem [29]. Consequently, in this subsection we develop a wood-size equalization algorithm to place those data items in the broadcast cycle.

Wood-Size Equalization WSE($\mathfrak{T}$):
// Let h:= the height of broadcast tree $\mathfrak{T}$,
// R:= the root node of tree $\mathfrak{T}$
// $\mathfrak{T}_L$ and $\mathfrak{T}_R$ denote the left and the right subtrees of $\mathfrak{T}$.
(1) **If** $h = 0$, then **return** $\mathfrak{T}$.
(2) **else** $(w_1, \ldots, w_{2^{h-1}})$:= WSE($\mathfrak{T}_L$),
(3) $(w_{2^{h-1}+1}, \ldots, w_{2^h})$:= WSE($\mathfrak{T}_R$),
(4) **return** *Root-Cutting*$(R, \beta_1, \ldots, \beta_{2^h})$.

Algorithm 2

The functionality of the wood-size equalization is to split broadcast tree $\mathfrak{T}$ of height h into 2^h pieces of broadcast wood $(w_1, \ldots, w_{2^h})$ with similar sizes. Actually, the wood-size equalization is a recurrence in structure. It splits the broadcast tree in a bottom-up manner. Given broadcast tree $\mathfrak{T}$, we first get $2^h/2$ broadcast woods by applying the wood-size equalization to the left subtree of $\mathfrak{T}$ and get the other $2^h/2$ broadcast woods from the right subtree of $\mathfrak{T}$. Afterwards, the *root-cutting* procedure permits the distribution of the data in the root to these woods such that each broadcast wood has a similar size. The wood-size equalization can be stated as follows.

Basically, the root-cutting procedure adopts a greedy strategy to divide the root node. More precisely, the root-cutting procedure iteratively splits the data a_i with the largest data size from the root and attaches it to the minimum-size wood w_c until all the data in the root are allocated. Thus, we can summarize the root-cutting procedure in Algorithms 2 and 3.

An example of the proposed wood-size equalization is illustrated in Figure 5. Consider broadcast tree $\mathfrak{T}$ of height 2 in Figure 3(d). In the beginning, we recursively apply wood-size equalization, resulting in the intermedium tree with four broadcast woods shown in Figure 5(a). After performing the root-cutting procedure, we get four individual broadcast woods as drawn in Figure 5(b).

Root-cutting procedure $(R, w_1, \ldots, w_v)$**:**
(1) Sort the data items in root R according to their sizes into a non-increasing order $a_1, \ldots, a_k$. (i.e., $S(a_i) \geq S(a_j)$ iff $1 \leq i \leq j \leq k$).
(2) **For** $i = 1$ to k
(3) Find the wood w_c with the smallest size.
(4) Split the data item a_i from the root and attach it to the top of wood w_c.
(5) **return** $(w_1, \ldots, w_v)$.

Algorithm 3

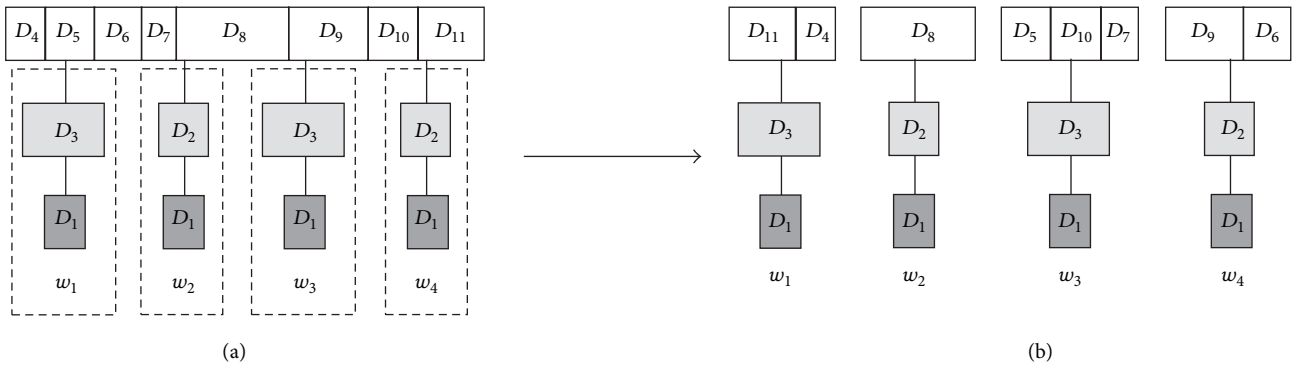

Figure 5: The scenario of the wood-size equalization.

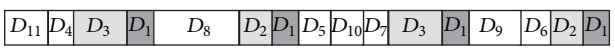

Figure 6: The generated broadcast program.

Upon completion of the wood-size equalization, one can obtain the broadcast cycle by sequentially broadcasting each wood from top to bottom. Therefore, in this case we can get the broadcast program as shown in Figure 6.

3.4. Complexity Analysis. In this subsection, the time complexity of the TAB algorithm is studied. Recall that the TAB algorithm contains three steps. The first step is the data-item reordering, which requires time complexity $O(N\log N)$ for sorting N data items. In addition, the second step, broadcast-tree construction, builds a broadcast tree having a height of at most τ. For each level, it takes at most $O(N)$ time to determine the fittest cutpoint. And then, it requires $O(2^\tau N)$ time to expand from a sapling to a full binary tree. Finally, the wood-size equalization is a recurrence in structure, and it requires a time complexity of $O(\tau N\log N + \tau 2^\tau N)$. Besides, because the value τ is relatively insignificant for the large value of N, we concluded that the proposed TAB algorithm takes only a time complexity of $O(N\log N)$ in total.

4. Optimum Bandwidth Allocation

In this section, we develop an analytical model to approximate the average access time of the proposed TAB algorithm (Theorem 9). Afterwards, this analytical model helps derive the optimum bandwidth allocation for our system architecture and minimize the average access time (Theorem 10). The details are described as follows.

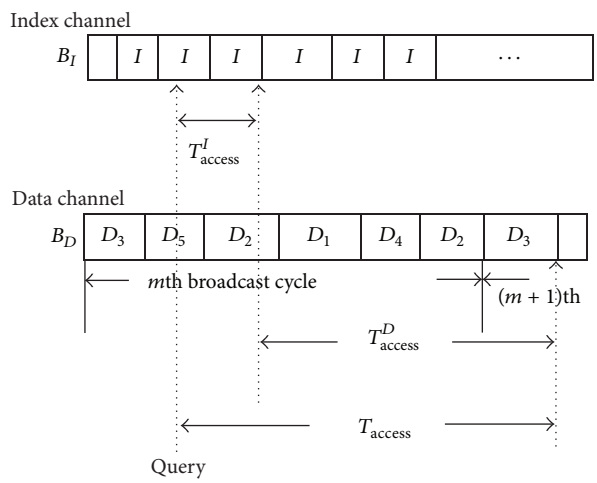

Figure 7: Sketch of the data-access time.

It can be seen from Figure 7 that the access time of the proposed TAB algorithm can be decomposed into two portions: *index-access time* and *data-access time*. When a user submits queries to mobile clients, the mobile device needs to read an index packet from the index channel. The time interval between the moment that the user sends a query and the moment that the mobile device gets one index packet is called the index-access time. Likewise, the data-access time is defined as the time interval between the moment that the

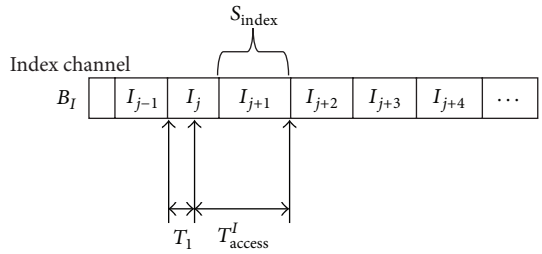

Figure 8: Sketch of the index-access time.

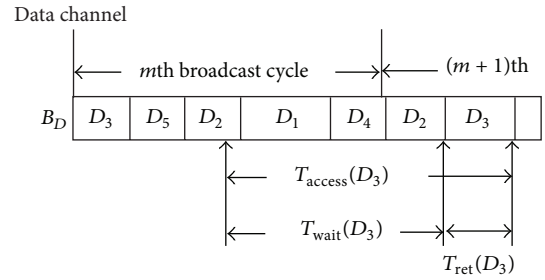

Figure 9: Sketch of the data-access time.

mobile client finishes the index packet and the moment that the mobile device obtains the target data packet. As a result, by the above definitions, we have the following lemma.

Lemma 2. *Let the random variable T_{access} denote the total access time of a query. And let the random variables T^I_{access} and T^D_{access} represent the index-access time and data-access time of the query, respectively. Thus, it is clear that*

$$T_{access} = T^I_{access} + T^D_{access}. \tag{5}$$

Therefore, according to Lemma 2, we know that the index-access time and data-access time should be calculated first to obtain the total access time. In order to get the index-access time, we consider the index channel as shown in Figure 8. It can be seen from Figure 8 that the index access time T^I_{access} is determined by the time the mobile device submits its query. Consequently, if we assume that a uniform distribution characterizes the duration of time extending from the starting point of the current index packets to the moment the mobile device submits its query [30], then the average index-access time can be arrived at by the following lemma.

Lemma 3. *Let S_{index} represent the size of each index packet and let B_I denote the bandwidth of the index channel. Assume that the interval between the starting point of the current index packet and the moment the mobile client submits its query follows a uniform distribution over $[0, S_{index}/B_I)$. Then the average index-access time can be formulated as*

$$E\left[T^I_{access}\right] = \int_0^{S_{index}/B_I} \left(2 \cdot \frac{S_{index}}{B_I} - t\right) \cdot \left(\frac{S_{index}}{B_I}\right)^{-1} dt. \tag{6}$$

Proof. Without loss of generality, we consider the case in which the mobile user submits a query during the jth index-packet broadcast time as in Figure 8. Let T_1 denote the random variable representing the time interval between the starting point of the jth index-packet broadcast cycle and the moment that the mobile device submits its query. Thus, it is clear that the index-access time T^I_{access} can be formulated as

$$T^I_{\text{access}} = \begin{cases} \dfrac{S_{\text{index}}}{B_I} & \text{if } T_1 = 0 \\ 2\dfrac{S_{\text{index}}}{B_I} - T_1 & \text{if } 0 < T_1 < \dfrac{S_{\text{index}}}{B_I}. \end{cases} \tag{7}$$

In addition, if we further assume that the random variable T_1 follows a uniform distribution over $[0, S_{\text{index}}/B_I)$, then the mean index-access time can be computed by

$$E\left[T^I_{\text{access}}\right] = \int_0^{S_{\text{index}}/B_I} \left(2 \cdot \frac{S_{\text{index}}}{B_I} - t\right) \cdot \left(\frac{S_{\text{index}}}{B_I}\right)^{-1} dt. \tag{8}$$

□

On the other hand, since each data item D_i has its own access probability $\Pr(D_i)$, the average data-access time can be expressed as a weighted summation of the average access time of all data items. In terms of mathematic form, the mean data-access time can be formulated as follows.

Lemma 4. *Suppose that the server database contains N data items $D_1, D_2, \ldots, D_N$ for broadcasting. Furthermore, let $\Pr(D_i)$ denote the access probability of the data item D_i, for $i = 1, 2, \ldots, N$. Then the expected value of the data-access time can be expressed by*

$$E\left[T^D_{access}\right] = \sum_{i=1}^{N} E\left[T_{access}(D_i)\right] \cdot \Pr(D_i), \tag{9}$$

where $T_{access}(D_i)$ stands for the random variable representing the access time of the specific data item D_i.

As a consequence, in order to obtain the average data-access time, we need to get the average access time for each data item first. Figure 9 depicts the sketch of the data item D_3's access time. As shown in Figure 9, the access time of an arbitrary data item D_i can be further decomposed into two parts: *waiting time* and *retrieval time*. The waiting time of an arbitrary data item D_i is defined as the time interval between the moment the mobile device gets an index packet and the moment the data channel starts to broadcast the target data item D_i. And the retrieval time represents the time interval during which the mobile equipment downloads the target data item. Thus, by the above definition, we have the following lemma.

Lemma 5. *Let $T_{wait}(D_i)$ denote the random variable representing the waiting time of the data item D_i and let $T_{ret}(D_i)$ denote the retrieval time of the data item D_i. Then we have*

$$T_{access}(D_i) = T_{wait}(D_i) + T_{ret}(D_i). \tag{10}$$

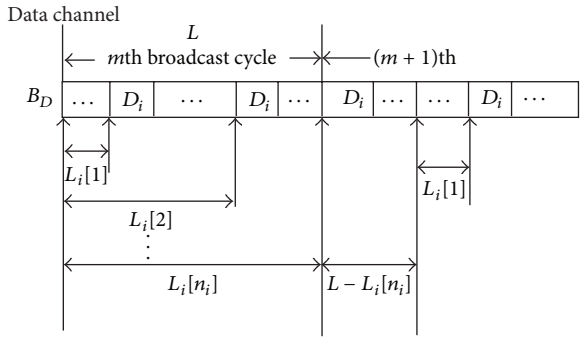

FIGURE 10: The broadcast deployment for data item D_i (L_i-function).

In addition, the retrieval time $T_{ret}(D_i)$ can be determined by its data size $S(D_i)$ divided by the data channel bandwidth B_D. That is,

$$T_{ret}(D_i) = \frac{S(D_i)}{B_D}. \tag{11}$$

On the other hand, it is not easy to derive the waiting time $T_{\text{wait}}(D_i)$ directly for some data items D_i because the proposed TAB algorithm broadcasts duplicates for those data items with high-access probability. Furthermore, the positions of the replicated data items in the broadcast cycle also determine the data items' waiting time. Therefore, in this paper, we introduce a specific $L_i[k]$-function to represent the position of the kth replicate of the data item D_i in the broadcast cycle.

Definition 6. The length L of a broadcast cycle is defined as the total number of data bits in this broadcast cycle. And the term $L_i[k]$ is defined as the total number of broadcasted bits before broadcasting the kth replicate of the data item D_i in a broadcast cycle.

Example 7. Take the broadcast cycle depicted in Figure 6 as an example. In this case, we have the equations $L_1[1] = S(D_{11}) + S(D_4) + S(D_3)$, $L_1[2] = L_1[1] + S(D_1) + S(D_8) + S(D_2)$, $L_1[3] = L_1[2] + S(D_1) + S(D_5) + S(D_{10}) + S(D_7) + S(D_3)$, and $L_1[4] = L_1[3] + S(D_1) + S(D_9) + S(D_6) + S(D_2)$. The length of the broadcast cycle L is equal to $L_1[4] + S(D_1)$.

As a result, we know that the $L_i[k]$-function can help describe any broadcast program accurately. Consider the case in which, after this work performs the proposed TAB algorithm, the data item D_i has n_i duplicates in a broadcast cycle, and these n_i duplicates are located at $L_i[1], L_i[2], \ldots, L_i[n_i]$, respectively (see Figure 10). Subsequently, the following theorem can yield the data item's waiting time $T_{\text{wait}}(D_i)$ relative to the proposed TAB scheme.

Theorem 8. *Let B_D be the bandwidth of the data channel. Suppose that the broadcast program obtained by performing the TAB algorithm is given in terms of the $L_i[k]$-function. Moreover, let the random variable T represent the time interval between the starting point of the current broadcast cycle and the moment that the mobile client starts to wait for the target data item. Then the random variable $T_{wait}(D_i)$ can be formulated as*

$$T_{wait}(D_i) = \begin{cases} \dfrac{L_i[1]}{B_D} - T & if\ 0 \le T < \dfrac{L_i[1]}{B_D} \\ \dfrac{L_i[k+1]}{B_D} - T & if\ \dfrac{L_i[k]}{B_D} \le T < \dfrac{L_i[k+1]}{B_D} \\ & for\ k = 1, 2, \ldots, n_i - 1 \\ \dfrac{L - T + L_i[1]}{B_D} & if\ \dfrac{L_i[n_i]}{B_D} \le T < \dfrac{L}{B_D}. \end{cases} \tag{12}$$

Besides, if we assume that the random variable T follows a uniform distribution over $[0, L/B_D)$, then the average waiting time $E[T_{wait}(D_i)]$ can be further simplified as

$$E[T_{wait}(D_i)] = \frac{1}{2B_D L}\left[(L - L_i[n_i] + L_i[1])^2 + \sum_{k=1}^{n_i - 1}(L_i[k+1] - L_i[k])^2\right]. \tag{13}$$

Proof. Without loss of generality, we assume that the mobile client starts to wait for the target data item D_i in the mth broadcast cycle as in Figure 10. Since the data D_i is broadcast n_i times during a broadcast cycle, the mobile client retrieves the nearest replicate of the target D_i according to the entry time the mobile client starts to wait. So with time T being the time at which the mobile device starts to wait, we now consider three cases for calculating the waiting time.

Case 1 ($0 \le T < L_i[1]/B_D$). In this case, the mobile client starts to wait before the first replicate of the target item in the mth broadcast cycle is broadcast. Therefore, the waiting time can be obtained by

$$T_{\text{wait}}(D_i) = \frac{L_i[1]}{B_D} - T. \tag{14}$$

Case 2 ($L_i[k]/B_D \le T < L_i[k+1]/B_D, k = 1, 2, \ldots, n_i - 1$). In this case, the mobile client retrieves the $(k+1)$th replicate of the target data item in the mth broadcast cycle. Thus, the waiting time can be computed by

$$T_{\text{wait}}(D_i) = \frac{L_i[k+1]}{B_D} - T. \tag{15}$$

Case 3 ($L_i[n_i]/B_D \le T < L/B_D$). In this case, all the replicates of the target data item in the mth broadcast cycle were broadcast when the mobile device began to wait. Thus, the mobile client would download the first replicate of the target data in the $(m+1)$th broadcast cycle. In other words, the waiting time can be formulated as

$$T_{\text{wait}}(D_i) = \frac{L - T + L_i[1]}{B_D}. \tag{16}$$

□

On the other hand, if we further assume that the random variable T satisfies a uniform distribution over $[0, L/B_D)$, then the expected value of the waiting time can be derived by

$$\begin{aligned} E\left[T_{\text{wait}}(D_i)\right] &= \int_0^{L_i[1]/B_D} \left(\frac{L_i[1]}{B_D} - t\right) \cdot \left(\frac{L}{B_D}\right)^{-1} dt \\ &\quad + \sum_{k=1}^{n_i-1} \int_{L_i[k]/B_D}^{L_i[k+1]/B_D} \left(\frac{L_i[k+1]}{B_D} - t\right) \cdot \left(\frac{L}{B_D}\right)^{-1} dt \\ &\quad + \int_{L_i[n_i]/B_D}^{L/B_D} \left(\frac{L - t + L_i[1]}{B_D}\right) \cdot \left(\frac{L}{B_D}\right)^{-1} dt \\ &= \frac{1}{2B_D L} \left[\left(L - L_i[n_i] + L_i[1]\right)^2 + \sum_{k=1}^{n_i-1} \left(L_i[k+1] - L_i[k]\right)^2 \right]. \end{aligned} \tag{17}$$

According to the above lemmas and theorems, the average access time can be computed as well. We now summarize the derivation of the average access time via the following theorem.

Theorem 9. *The average access time $E[T_{access}]$ can be derived by*

$$\begin{aligned} E\left[T_{access}\right] &= \frac{3S_{index}}{2B_I} + \sum_{i=1}^{N} \Pr\left(D_i\right) \cdot \left\{ \frac{1}{2B_D L} \left[\left(L - L_i[n_i] + L_i[1]\right)^2 + \sum_{k=1}^{n_i-1} \left(L_i[k+1] - L_i[k]\right)^2 \right] + \frac{S(D_i)}{B_D} \right\}. \end{aligned} \tag{18}$$

Proof. Based on Lemma 2, it is clear that

$$E\left[T_{\text{access}}\right] = E\left[T_{\text{access}}^I\right] + E\left[T_{\text{access}}^D\right]. \tag{19}$$

Furthermore, after applying Lemmas 4 and 5 to the above equation, we get

$$E\left[T_{\text{access}}\right] = E\left[T_{\text{access}}^I\right] + \sum_{i=1}^{N} \Pr\left(D_i\right) \cdot \left(E\left[T_{\text{wait}}(D_i)\right] + \frac{S(D_i)}{B_D} \right). \tag{20}$$

Finally, Lemma 3 and Theorem 8 permit us to calculate the average access time $E[T_{\text{access}}]$ as follows:

$$\begin{aligned} &\frac{3S_{\text{index}}}{2B_I} + \sum_{i=1}^{N} \Pr\left(D_i\right) \cdot \left\{ \frac{1}{2B_D L} \left[\left(L - L_i[n_i] + L_i[1]\right)^2 + \sum_{k=1}^{n_i-1} \left(L_i[k+1] - L_i[k]\right)^2 \right] + \frac{S(D_i)}{B_D} \right\}. \end{aligned} \tag{21}$$

From Theorem 9, we can not only estimate the average access time of a query for any broadcast program, but also determine the optimum bandwidth allocation for both the index channel and the data channel. Consider the case in which the overall channel bandwidth for data broadcasting is B. Then, for the proposed TAB algorithm to achieve the minimum access time, the optimum bandwidth allocation is given by the following theorem. □

Theorem 10. *Let B denote the total bandwidth for data broadcasting. Then the optimum bandwidth settings necessary for index channel B_I^{opt} and data channel B_D^{opt} to achieve the minimum average access time can be formulated as*

$$B_I^{opt} = \frac{\sqrt{3S_{index}}}{\sqrt{3S_{index}} + \sqrt{2\xi}} B, \qquad B_D^{opt} = \frac{\sqrt{2\xi}}{\sqrt{3S_{index}} + \sqrt{2\xi}} B, \tag{22}$$

where

$$\xi = \sum_{i=1}^{N} \Pr\left(D_i\right) \cdot \left\{ \frac{1}{2L} \cdot \left[\left(L - L_i[n_i] + L_i[1]\right)^2 + \sum_{k=1}^{n_i-1} \left(L_i[k+1] - L_i[k]\right)^2 \right] + S(D_i) \right\}. \tag{23}$$

Proof. According to Theorem 9, the average access time $E[T_{\text{access}}]$ is equal to

$$E\left[T_{\text{access}}\right] = \frac{3S_{\text{index}}}{2B_I} + \frac{\xi}{B_D}. \tag{24}$$

Denote the estimation function $\Phi(B_I, B_D)$ by

$$\Phi\left(B_I, B_D\right) = \frac{3S_{\text{index}}}{2B_I} + \frac{\xi}{B_D}. \tag{25}$$

As a consequence, to minimize the average access time $E[T_{\text{access}}]$, we need to choose the values B_I and B_D to minimize the estimation function $\Phi(B_I, B_D)$ subject to the constraint $B_I + B_D = B$.

Table 3: Parameter setting.

Parameters	Values
The number of data items (N)	50~1000
Bandwidth (B)	80 KB/sec
Index packet size (S_{index})	128 bytes
The sizes of data items ($S(D_i)$)	Normal distribution (mean: 50~150 KB, variance: 900 KB^2)
The access probabilities ($\Pr(D_i)$)	Zipf distribution (θ = 0.5~1.5)
The number of requests	5000
Maximum height of broadcast tree	3

Assume that $\Gamma(B_I) = \Phi(B_I, B - B_I)$. Then it is clear that the optimum bandwidth of the index channel B_I^{opt} satisfies the equation

$$\left.\frac{d\Gamma(B_I)}{dB_I}\right|_{B_I=B_I^{\text{opt}}} = 0. \quad (26)$$

After substituting this into the estimation function, we have

$$\left.\frac{d\Gamma(B_I)}{dB_I}\right|_{B_I=B_I^{\text{opt}}} = \left.\left(-\frac{3S_{\text{index}}}{2B_I^2} + \frac{\xi}{(B-B_I)^2}\right)\right|_{B_I=B_I^{\text{opt}}} = 0. \quad (27)$$

That is,

$$B_I^{\text{opt}} = \frac{\sqrt{3S_{\text{index}}}}{\sqrt{3S_{\text{index}}} + \sqrt{2\xi}} B. \quad (28)$$

Also, the optimum bandwidth of the data channel B_D^{opt} can be arrived at through the following equation:

$$B_D^{\text{opt}} = B - B_I^{\text{opt}} = \frac{\sqrt{2\xi}}{\sqrt{3S_{\text{index}}} + \sqrt{2\xi}} B. \quad (29)$$

□

5. Performance Evaluation

5.1. Simulation Environment. In order to assess the performance of the proposed system architecture, we conducted several experiments. Table 3 shows the parameter settings in our experiments. We assumed that the number of total data records for broadcasting varied from 50 to 100. And each data item contained two attributes: data size and data-access frequency.

In our experiments, the data sizes followed a normal distribution with the mean varying from 50 KB to 150 KB and a variance of 900 KB^2. The modeling of the data-access probabilities rested on the Zipf distribution with the parameter θ [31]. In other words, the probability of the data item D_i was assumed to be

$$\Pr(D_i) = \frac{(1/i)^\theta}{\sum_{j=1}^{N} (1/j)^\theta}, \quad (30)$$

where the value of skew factor θ ranged from 0.5 to 1.5.

We did not employ any indexing technology for the index channel in our system, and in this way we could realize the actual effects of the proposed method. The index packet size was assumed to be 128 bytes. Further, the total available bandwidth including the index and data channel was set to 80 KB/sec [3].

In addition to the proposed system architecture, we implemented a plain broadcast, broadcast disks, and an exponential index scheme for comparison. In line with the simulation model in [1], the broadcast-disk technology was implemented with three broadcast disks. And the relative frequencies between these disks were dominated by parameter Δ. More precisely, the broadcast frequency of the disk i was determined by $ref\_freq(i) = (3-i)\Delta + 1$. In the experiments, we considered three kinds of broadcast disks schemes: Δ = 1, 2, and 3.

For each experiment, we generated five different datasets, each one containing 50~100 data records for broadcasting. In addition, for each data set, we generated 5,000 queries and calculated the corresponding access time and tune-in time to evaluate access efficiency and power conservation. The interarrival time of queries followed an exponential distribution with an arrival rate of λ = 1. The simulator and query generator were coded in MATLAB.

5.2. Experimental Results. This work applies average tune-in time and average access time as the performance measurement. It allows devices to power on when they need to access data so that these two values are lower than the ones gained by the other method. It means that this device can stay in sleep mode longer and save more energy.

Figure 11 shows the average access time and tune-in time of different schemes with the number of data items varying from 50 to 100. In this experiment, we considered the case in which the Zipf parameter was set at 0.9 and the data-size generation process was a normal distribution with a mean of 100 KB and a variance of 900 KB^2.

It can be seen from Figure 11(a) that even though we sacrificed some bandwidth to broadcast index packets, our mechanism still achieved a lower access time than the broadcast disks did. Furthermore, as presented in Figure 11(a), this phenomenon became more prominent as the number of data items increased. Figure 11(b) shows the effects of our system on power conservation. As shown in Figure 11(b), the average tune-in time of our mechanism was lower than that of the exponential-index scheme. This finding demonstrates the importance of efficient bandwidth allocation in power conservation.

Figure 12 presents our comparison of the access latency and tune-in time in various average data sizes. Without loss of generality, we considered the number of data records to be set at 75 and the Zipf parameter at 0.9 in this experiment. Furthermore, the data-size generation process followed a normal distribution with a mean ranging from 50 KB to 100 KB.

As shown in Figure 12(a), the average access latency of all schemes increased as the average data size increased. Nevertheless, our mechanism consistently outperformed all the other schemes for various data sizes. In addition, the

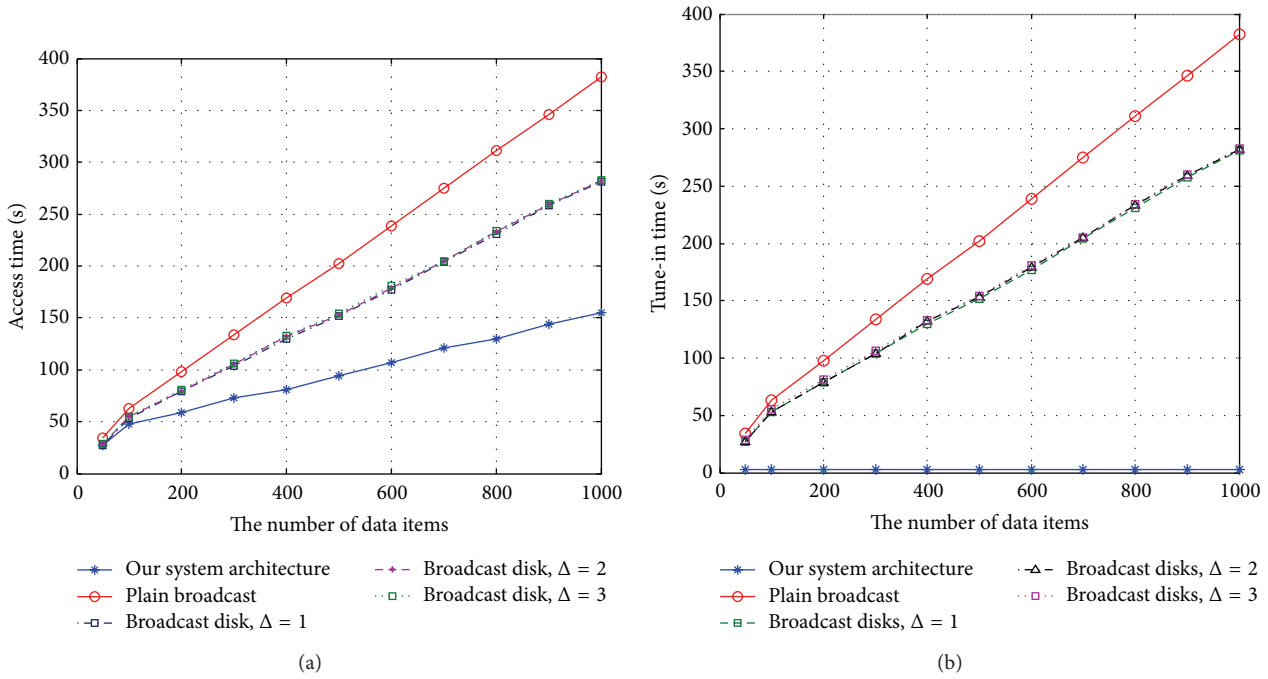

Figure 11: The average access time and tune-in time for various data-item numbers.

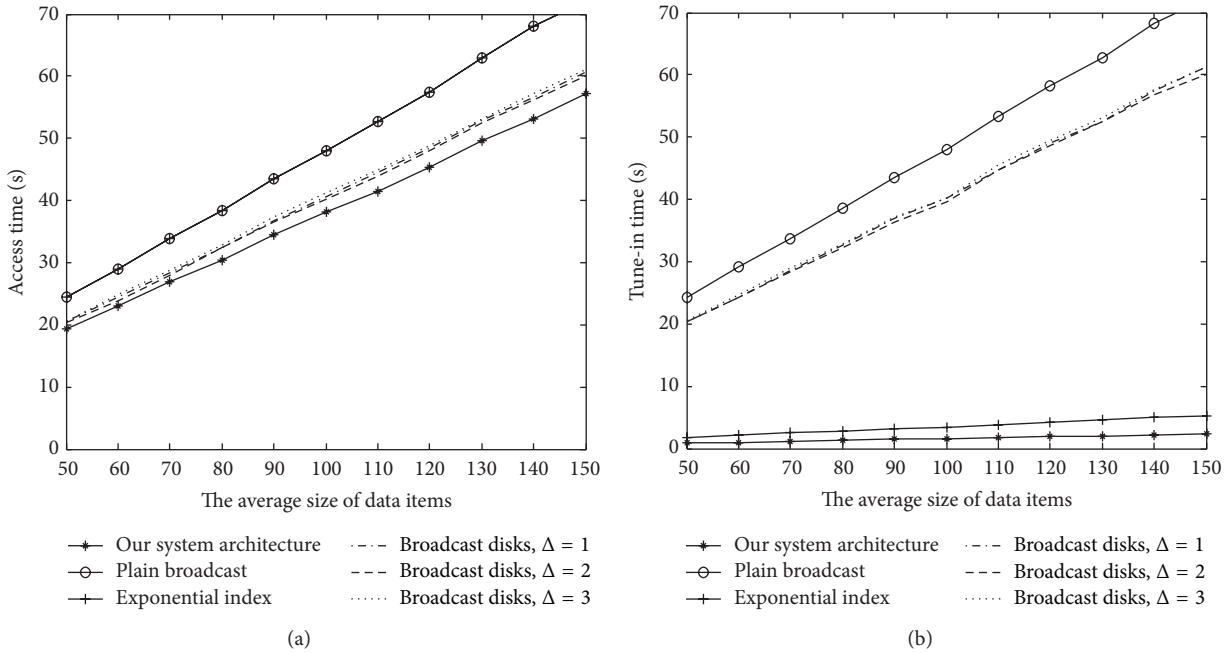

Figure 12: Performance comparison in various average data sizes.

slope of our mechanism proved to be smaller than that of the broadcast disks. A similar condition appeared in our comparison of the tune-in time. As depicted in Figure 12(b), because our system used the optimum bandwidth allocation, our mechanism could achieve a better performance in power conservation than the exponential index scheme.

Regarding the effects of the skewness of the data-access probabilities, Figure 13 depicts our comparison of the average access time and tune-in time in various Zipf parameters. In this experiment, the Zipf parameter representing the skewness of data-access frequencies varied from 0.5 to 1.5. The number of data records was 100 and the data size had a normal distribution with a mean of 100 KB and a variance of 900 KB^2.

It can be seen from Figure 13(a) that our mechanism significantly outperforms all the other schemes. Furthermore, the performance gain of our mechanism becomes more and

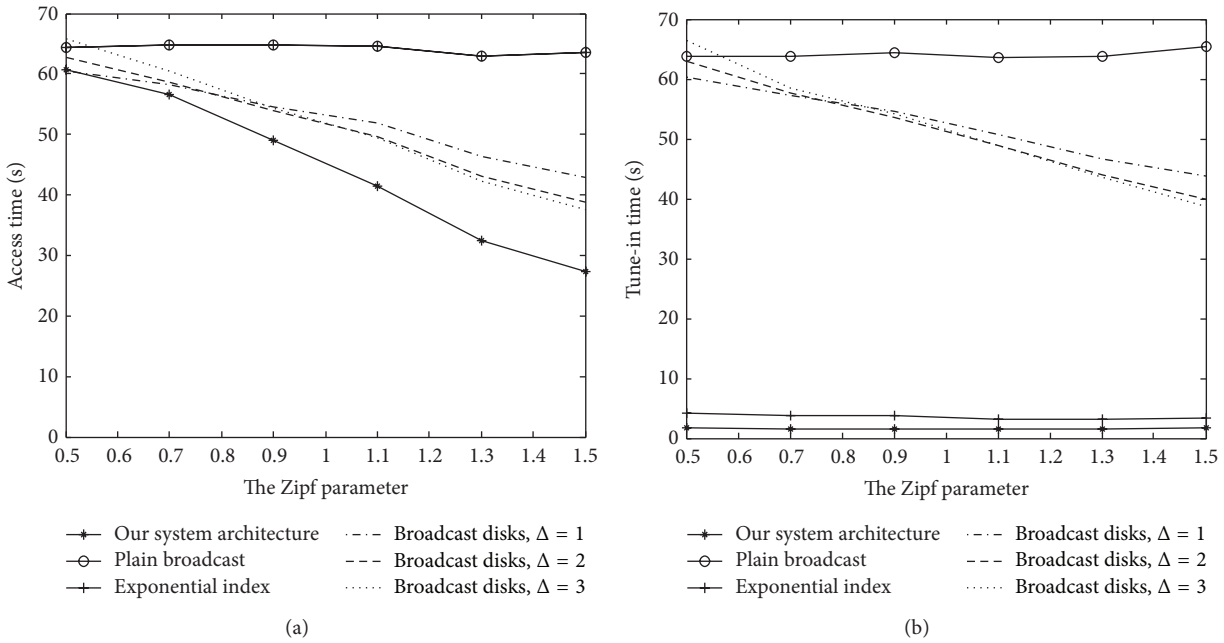

Figure 13: Performance comparison in various skew factors.

more conspicuous as the value of the skew factor increases. This indicates that an efficient determination of the data-broadcast frequency is critical, especially the more skewed the data access is. In terms of power conservation, Figure 13(b) shows that the skew factor only slightly affects the tune-in time of our mechanism and the exponential index scheme.

Regarding the accuracy of our approximation model, Figure 14 presents our comparison between the analytic results and the experimental results. We obtained each point depicted in the curve of approximation by calculating the derived access time in Theorem 9. Each point shown in the curve of the simulation results represents the average access time of 5,000 data queries on various parameters. Again, Figure 14 shows that the curve of our approximation is close to that of the numerical results.

In conclusion, even though our system releases some bandwidth to broadcast index packets, our experimental results show that our mechanism exhibits better access latency than the plain broadcast and broadcast disks scheme do, especially when the data-access probabilities are skewed. In terms of power conservation, it is clear that our system can reduce much more tune-in time if a proper indexing technology is applied to our index channel. In addition, the numerical results of our experiments confirm the accuracy of the proposed approximation model.

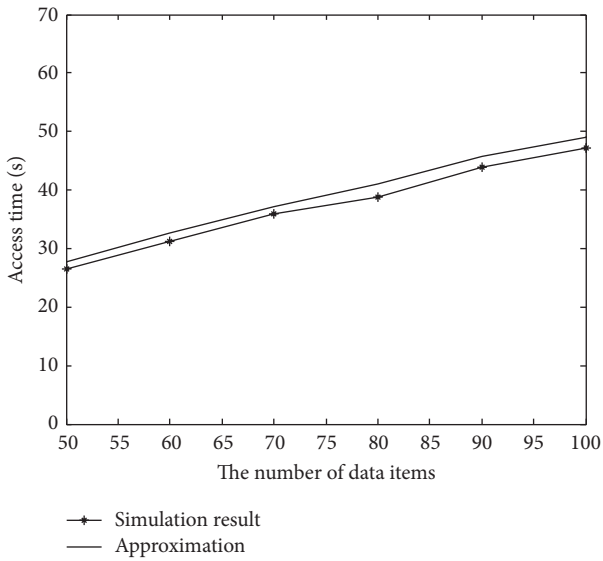

Figure 14: Average access time for experimental and analytical results.

6. Conclusion

Data broadcasting involves important data dissemination technology for accessing mobile services in wireless networks. In general, there are two main approaches to data broadcast, namely, push-based broadcast and on-demand broadcast [32]. Mobile Internet and mobile services that make use of mobile data are increasingly popular [33–35]. Among others, access efficiency and power conservation are two critical performance indexes for assessing the effectiveness of wireless communication systems. In this paper, we present a TAB algorithm to reduce the response time of mobile clients' requests. We provide an analytical model to measure the expected access latency of the generated broadcast program. This analytical model helps formulate the optimum bandwidth allocation for index and data channels. From the experimental results, it can be seen that our mechanism outperforms the existing data-broadcast schemes in terms of access time. Moreover, the optimum bandwidth

allocation also brings about a significant improvement in energy conservation. Based on these advantages, it can be seen that the proposed mechanism is scalable and can feasibly increase the efficiency of data dissemination in broadcast-based systems.

Conflict of Interests

The authors declare that there is no conflict of interests.

Acknowledgments

The authors thank the National Science Council of Taiwan for funding this research (Project no. NSC 102-2218-E-268-001).

References

[1] S. T. Cheng, J. P. Liu, J. L. Kao, and C. M. Chen, "A New Framework for Mobile Web Services," in *Proceedings of the IEEE Symposium on Applications and the Internet*, pp. 218–222, 2002.

[2] X. Yang, A. Bouguettaya, B. Medjahed, H. Long, and W. He, "Organizing and Accessing Web Services on Air," *IEEE Transactions on Systems, Man, and Cybernetics A: Systems and Humans*, vol. 33, no. 6, pp. 742–757, 2003.

[3] T. Imielinski and B. R. Badrinath, "Mobile wireless computing challenges in data management," *Communications of the ACM*, vol. 37, no. 10, pp. 18–28, 1994.

[4] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Data on air: Organization and access," *IEEE Transactions on Knowledge and Data Engineering*, vol. 9, no. 3, pp. 353–372, 1997.

[5] S. Acharya, R. Alonso, M. Franklin, and S. Zdonik, "Broadcast disks: data management for asymmetric communications environments," in *Proceedings of the ACM SIGMOD Conference on Management of Data*, pp. 199–210, May 1995.

[6] J. L. Huang and M. S. Chen, "Dependent data broadcasting for unordered queries in a multiple channel mobile environment," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1143–1156, 2004.

[7] P. Sumari, R. M. Darus, and H. Kamarulhaili, "Data organization for broadcasting in mobile computing," in *Proceedings of the International Conference on Geometric Modeling and graphics*, pp. 49–54, July 2003.

[8] S. Hameed and N. H. Vaidya, "Efficient algorithms for scheduling data broadcast," *Wireless Networks*, vol. 5, no. 3, pp. 183–193, 1999.

[9] G. Lee and S.-C. Lo, "Broadcast data allocation for efficient access of multiple data items in mobile environments," *Mobile Networks and Applications*, vol. 8, no. 4, pp. 365–375, 2003.

[10] W. Sun, W. Shi, B. Shi, and Y. Yu, "A cost-efficient scheduling algorithm of on-demand broadcasts," *Wireless Networks*, vol. 9, no. 3, pp. 239–247, 2003.

[11] K. F. Jea and M. H. Chen, "A data broadcast scheme based on prediction for the wireless environment," in *Proceedings of the 9th International Conference Parallel and Distributed Systems (ICPADS '02)*, December 2002.

[12] H. P. Hung and M. S. Chen, "On exploring channel allocation in the diverse data broadcasting environment," in *Proceedings of the 25th IEEE International Conference on Distributed Computing Systems*, pp. 729–738, June 2005.

[13] L. S. Juhn and L. M. Tseng, "Fast data broadcasting and receiving scheme for popular video service," *IEEE Transactions on Broadcasting*, vol. 44, no. 1, pp. 100–105, 1998.

[14] W. Chung, T. J. Endres, and C. D. Long, "A data broadcasting system expanding the information capacity of existing analog communication systems," *IEEE Transactions on Broadcasting*, vol. 51, no. 2, pp. 180–190, 2005.

[15] T. Imielinski, S. Viswanathan, and B. R. Badrinath, "Energy efficient indexing on air," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 25–36, May 1994.

[16] G. Herman, G. Gopal, K. Lee, and A. Weinrib, "The datacycle architecture for very high throughput database systems," in *Proceedings of the ACM SIGMOD Conference on Management of Data*, May 1987.

[17] X. Yang and A. Bouguettaya, "Adaptive data access in broadcast-based wireless environments," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 3, pp. 326–338, 2005.

[18] L. Yin and G. Cao, "Adaptive power-aware prefetch in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 3, no. 5, pp. 1648–1658, 2004.

[19] M. A. Viredaz, L. S. Brakmo, and W. R. Hamburgen, "Energy management on handheld devices," *ACM Queue*, pp. 44–52, 2003.

[20] K. L. Tan and B. C. Ooi, *Data Dissemination in Wireless Computing Environments*, Kluwer Academic, New York, NY, USA, 2000.

[21] J. Xu, W.-C. Lee, and X. Tang, "Exponential index: a parameterized distributed indexing scheme for data on air," in *Proceedings of the 2nd ACM/USENIX International Conference on Mobile Systems, Applications and Services (MobiSys '04)*, pp. 153–164, June 2004.

[22] M. Chen, P. S. Yu, and K. Wu, "Indexed sequential data broadcasting in wireless mobile computing," in *Proceedings of the 17th International Conference Distributed Computing Systems*, May 1997.

[23] M.-S. Chen, K.-L. Wu, and P. S. Yu, "Optimizing index allocation for sequential data broadcasting in wireless mobile computing," *IEEE Transactions on Knowledge and Data Engineering*, vol. 15, no. 1, pp. 161–173, 2003.

[24] N. Shivakumar and S. Venkatasubramanian, "Efficient indexing for broadcast based wireless systems," *Mobile Networks and Applications*, vol. 1, no. 4, pp. 433–446, 1996.

[25] Q. Hu, W. C. Lee, and D. L. Lee, "Indexing techniques for wireless data broadcast under data clustering and scheduling," in *Proceedings of the 8th International Conference on Information Knowledge Management*, pp. 351–358, November 1999.

[26] W. C. Lee and D. L. Lee, "Using signature techniques for information filtering in wireless and mobile environments," *Distributed and Parallel Databases*, vol. 4, no. 3, pp. 205–227, 1996.

[27] Y. Huang, P. Sistla, and O. Wolfson, "Data replication for mobile computers," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pp. 13–24, May 1994.

[28] O. Wolfson and A. Milo, "Multicast policy and its relationship to replicated data placement," *ACM Transactions on Database Systems*, vol. 16, no. 1, pp. 181–205, 1991.

[29] M. Sipser, *Introduction To the Theory of Computation*, Course Technology.

[30] R. D. Yates and D. J. Goodman, *Probability and Stochastic Processes*, John Wiley & Sons, New York, NY, USA, 2004.

[31] G. K. Zipf, *Human Behaviour and the Principle of Least Effort*, Addison-Wesley, New York, NY, USA, 1949.

[32] H. Wang, Y. Xiao, and L. Shu, "Scheduling periodic continuous queries in real-time data broadcast environments," *IEEE Transactions on Computers*, vol. 61, no. 9, pp. 1325–1340, 2012.

[33] A. Molnar and C. H. Muntean, "Cost-oriented adaptive multimedia delivery," *IEEE Transactions on Broadcasting*, vol. 59, no. 3, pp. 484–499, 2013.

[34] G. J. Horng, C. H. Wang, S. T. Cheng, C. W. Hsu, and S. F. Su, "Tree-based adaptive broadcasting of bandwidth allocation for vehicle ad hoc networks," in *Proceedings of the 12th IEEE International Conference on High Performance Computing and Communications (HPCC '10)*, pp. 391–397, September 2010.

[35] Y. L. Lai and J. R. Jiang, "Pricing resources in LTE networks through multiobjective optimization," *The Scientific World Journal*, vol. 2014, Article ID 394082, 9 pages, 2014.